

The opcode tables in this chapter are provided to aid in interpreting Intel Architecture object code. The instructions are divided into three encoding groups: 1-byte opcode encodings, 2-byte opcode encodings, and escape (floating-point) encodings. The 1- and 2-byte opcode encodings are used to encode integer, system, and MMX instructions. The opcode maps for these instructions are given in Tables 37-1 through 37-6. “One-Byte Opcode Integer Instructions”, “Two-Byte Opcode Integer Instructions”, and “Opcode Extensions for One- and Two-byte Opcodes” give instructions for interpreting 1- and 2-byte opcode maps. The escape encodings are used to encode floating-point instructions. The opcode maps for these instructions are given in Tables 37-7 through 37-16. “Escape Opcode Instructions” gives instructions for interpreting the escape opcode maps.

See “Instruction Formats and Encodings”, for detailed information on the ModR/M byte, register values, and the various addressing forms.

37.1 Key to Abbreviations

Operands are identified by a two-character code of the form Zz. The first character, an uppercase letter, specifies the addressing method; the second character, a lowercase letter, specifies the type of operand.

37.1.1 Codes for Addressing Method

The following abbreviations are used for addressing methods:

- A** Direct address. The instruction has no ModR/M byte; the address of the operand is encoded in the instruction; and no base register, index register, or scaling factor can be applied, for example, far JMP (EA).
- C** The reg field of the ModR/M byte selects a control register, for example, MOV (0F20, 0F22).
- D** The reg field of the ModR/M byte selects a debug register, for example, MOV (0F21,0F23).
- E** A ModR/M byte follows the opcode and specifies the operand. The operand is either a general-purpose register or a memory address. If it is a memory address, the address is computed from a segment register and any of the following values: a base register, an index register, a scaling factor, a displacement.
- F** EFLAGS Register.
- G** The reg field of the ModR/M byte selects a general register, for example, AX (000).
- I** Immediate data. The operand value is encoded in subsequent bytes of the instruction.
- J** The instruction contains a relative offset to be added to the instruction pointer register, for example, JMP short, LOOP.
- M** The ModR/M byte may refer only to memory, for example, BOUND, LES, LDS, LSS, LFS, LGS, CMPXCHG8B.
- O** The instruction has no ModR/M byte; the offset of the operand is coded as a word or double word (depending on address size attribute) in the instruction. No base register, index register, or scaling factor can be applied, for example, MOV (A0–A3).

P	The reg field of the ModR/M byte selects a packed quadword MMX register.
Q	An ModR/M byte follows the opcode and specifies the operand. The operand is either an MMX register or a memory address. If it is a memory address, the address is computed from a segment register and any of the following values: a base register, an index register, a scaling factor, and a displacement.
R	The mod field of the ModR/M byte may refer only to a general register, for example, MOV (0F20-0F24, 0F26).
S	The reg field of the ModR/M byte selects a segment register, for example, MOV (8C,8E).
T	The reg field of the ModR/M byte selects a test register, for example, MOV (0F24,0F26).
X	Memory addressed by the DS:SI register pair (for example, MOVS, OUTS, or LODS).
Y	Memory addressed by the ES:DI register pair (for example, MOVS, INS, or STOS).

37.1.2 Codes for Operand Type

The following abbreviations are used for operand types:

a	Two one-word operands in memory or two double-word operands in memory, depending on operand size attribute (used only by the BOUND instruction).
b	Byte, regardless of operand-size attribute.
c	Byte or word, depending on operand-size attribute.
d	Doubleword, regardless of operand-size attribute.
p	32-bit or 48-bit pointer, depending on operand size attribute.
q	Quadword, regardless of operand-size attribute.
s	6-byte pseudo-descriptor.
v	Word or doubleword, depending on operand-size attribute.
w	Word, regardless of operand-size attribute.

37.1.3 Register Codes

When an operand is a specific register encoded in the opcode, the register is identified by its name (for example, AX, CL, or ESI). The name of the register indicates whether the register is 32, 16, or 8 bits wide. A register identifier of the form eXX is used when the width of the register depends on the operand size attribute. For example, eAX indicates that the AX register is used when the operand size attribute is 16, and the EAX register is used when the operand size attribute is 32.

37.2 One-Byte Opcode Integer Instructions

The opcode map for 1-byte opcodes are shown in Table 37-1. For 1-byte opcodes, the instruction and its operands can be determined from the hexadecimal opcode. For example, the opcode 030500000000H for an ADD instruction can be interpreted from the 1-byte opcode map in Table 37-1 as follows. The first digit (0) of the opcode indicates the row and the second digit (3) indicates the column in the opcode map table, which points to ADD instruction with operand types Gv and Ev. The first operand (type Gv) indicates a general register that is a word or doubleword depending on the operand-size attribute. The second operand (type Ev) indicates that a ModR/M byte follows that specifies whether the operand is a word or doubleword general-purpose register or a memory address. The ModR/M byte for this instruction is 05H, which indicates that a 32-bit displacement

follows (00000000H). The reg(opcode portion of the ModR/M byte (bits 3 through 5) is 000 indicating the EAX register. Thus, it can be determined that the instruction for this opcode is ADD EAX, mem_op and the offset of mem_op is 00000000H.

Some 1- and 2-byte opcodes point to “group” numbers. These group numbers indicate that the instruction uses the reg(opcode bits in the ModR/M byte as an opcode extension (see “Opcode Extensions for One- and Two-byte Opcodes”).

37.3 Two-Byte Opcode Integer Instructions

Instructions that begin with 0FH can be found in the two-byte opcode map given in Table 37-3. Here, the second opcode byte is used to reference a row and column in the Table. For example, the opcode 0FA4050000000003H is located on the first page of the two-byte opcode map in row A, column 4, which points to an SHLD instruction with the operands Ev, Gv, and Ib. The Ev, Gv, and Ib operands are interpreted as follows. The first operand (Ev type) indicates that a ModR/M byte follows the opcode to specify a word or doubleword operand. The second operand (Gv type) indicates that the reg field of the ModR/M byte selects a general-purpose register. The third operand (Ib type) indicates that immediate data is encoded in the subsequent byte of the instruction.

The third byte of the opcode (05H) is the ModR/M byte. The mod and opcode/reg fields indicate that a 32-bit displacement follows and that the EAX register is the source.

Table 37-1. One-Byte Opcode Map¹ (0–7)

	0	1	2	3	4	5	6	7
0	ADD						PUSH	POP
	Eb,Gb	Ev,Gv	Gb,Eb	Gv,Ev	AL,Ib	eAX,Iv	ES	ES
1	ADC						PUSH	POP
	Eb,Gb	Ev,Gv	Gb,Eb	Gv,Ev	AL,Ib	eAX,Iv	SS	SS
2	AND						SEG	DAA
	Eb,Gb	Ev,Gv	Gb,Eb	Gv,Ev	AL,Ib	eAX,Iv	=ES	
3	XOR						SEG	AAA
	Eb,Gb	Ev,Gv	Gb,Eb	Gb,Ev	AL,Ib	eAX,Iv	=SS	
4	INC general register							
	eAX	eCX	eDX	eBX	eSP	eBP	eSI	eDI
5	PUSH general register							
	eAX	eCX	eDX	eBX	eSP	eBP	eSI	eDI
6	PUSHA	POPA	BOUND	ARPL	SEG	SEG	Operand	Address
	PUSHAD	POPAD	Gv,Ma	Ew,Gw	=FS	=GS	Size	Size
7	Short-displacement jump on condition (Jb)							

Table 37-1. One-Byte Opcode Map¹ (0–7)

	0	1	2	3	4	5	6	7
	JO	JNO	JB/JNAE/ JC	JNB/JAE/ JNC	JZ	JNZ	JBE	JNBE
8	Immediate Group 1 ²				TEST		XCHG	
	Eb,Ib	Ev,Iv	Ev,Ib	Eb,Ib	Eb,Gb	Ev,Gv	Eb,Gb	Ev,Gv
9	NOP	XCHG word or double-word register with eAX						
		eCX	eDX	eBX	eSP	eBP	eSI	eDI
A	MOV				MOVSB	MOVSW	CMPSB	CMPSW
	AL,Ob	eAX,Ov	Ob,AL	Ov,eAX	Xb,Yb	Xv,Yv	Xb,Yb	Xv,Yv
B	MOV immediate byte into byte register							
	AL	CL	DL	BL	AH	CH	DH	BH
C	Shift Group 2a ²		RET near		LES	LDS	MOV	
	Eb,Ib	Ev,Ib	Iw		Gv,Mp	Gv,Mp	Eb,Ib	Ev,Iv
D	Shift Group 2 ²				AAM	AAD		XLAT
	Eb,1	Ev,1	Eb,CL	Ev,CL				
E	LOOPN	LOOPE	LOOP	JCXZ/ JECXZ	IN		OUT	
	Jb	Jb	Jb	Jb	AL,Ib	eAX,Ib	Ib,AL	Ib,eAX
F	LOCK		REPNE	REP	HLT	CMC	Unary Group 3 ²	
				REPE			Eb	Ev

Table 37-2. One-Byte Opcode Map (8–F)

	8	9	A	B	C	D	E	F
0	OR					PUSH	2-byte	
	Eb,Gb	Ev,Gv	Gb,Eb	Gv,Ev	AL,Ib	eAX,Iv	CS	Escape
1	SBB					PUSH	POP	
	Eb,Gb	Ev,Gv	Gb,Eb	Gv,Ev	AL,Ib	eAX,Iv	DS	DS
2	SUB					SEG	DAS	
	Eb,Gb	Ev,Gv	Gb,Eb	Gv,Ev	AL,Ib	eAX,Iv	=CS	
3	CMP	SEG	AAS					
	Eb,Gb	Ev,Gv	Gb,Eb	Gv,Ev	AL,Ib	eAX,Iv	=DS	
4	DEC General-Purpose Register							

Table 37-2. One-Byte Opcode Map (8–F)

	8	9	A	B	C	D	E	F
	eAX	eCX	eDX	eBX	eSP	eBP	eSI	eDI
5	POP Into General-Purpose Register							
	eAX	eCX	eDX	eBX	eSP	eBP	eSI	eDI
6	PUSH	IMUL	PUSH	IMUL	INSB	INSW/D	OUTSB	OUTSW/D
	lv	Gv,Ev,lv	lb	Gv,Ev,lb	Yb,DX	Yv,DX	Dx,Xb	DX,Xv
7	Short-Displacement Jump on Condition (Jb)							
	JS	JNS	JP	JNP	JL	JNL	JLE	JNLE
8	MOV				MOV	LEA	MOV	POP
	Eb,Gb	Ev,Gv	Gb,Eb	Gv,Ev	Ew,Sw	Gv,M	Sw,Ew	Ev
9	CBW/ CWDE	CWD/CDQ	CALL	WAIT	PUSHF	POP	SAHF	LAHF
			Ap		Fv	Fv		
A	TEST		STOSB	STOSW/D	LODSB	LODSW/D	SCASB	SCASW/D
	AL,lb	eAX,lv	Yb,AL	Yv,eAX	AL,Xb	eAX,Xv	AL,Yb	eAX,Yv
B	MOV Immediate Word or Double Into Word or Double Register							
	eAX	eCX	eDX	eBX	eSP	eBP	eSI	eDI
C	ENTER	LEAVE	RET far	RET far	INT	INT	INTO	IRET
	lw, lb		lw		3	lb		
D	ESC (Escape to Coprocessor Instruction Set)							
E	CALL	JMP			IN		OUT	
	Jv	Jv	Ap	Jb	AL,DX	eAX,DX	DX,AL	DX,eAX
F	CLC	STC	CLI	STI	CLD	STD	INC/DEC	INC/DEC
							Group 4 ²	Group 5 ²

NOTES:

1. All blanks in the opcode map are reserved and should not be used. Do not depend on the operation of these undefined opcodes.
2. Bits 5, 4, and 3 of ModR/M byte used as an opcode extension (see “Opcode Extensions for One- and Two-byte OpCodes”).

The next part of the SHLD opcode is the 32-bit displacement for the destination memory operand (00000000H), which is followed by the immediate byte representing the count of the shift (03H). By this breakdown, it has been shown that the opcode 0FA4050000000003H represents the instruction: SHLD DS:00000000H, EAX, 3.

Table 37-3. Two Byte Opcode Map (0–7) (First byte is 0FH)¹

		0	1	2	3	5	6	7
0	Group 6 ²		LAR Gv,Ew	LSL Gv,Ew			CLTS	
1								
2	MOV Rd,Cd	MOV Rd,Dd	MOV Cd,Rd	MOV Dd,Rd				
3	WRMSR	RDTSC	RDMSR	RDPMC				
4	CMOVO Gv, Ev	CMOVNO Gv, Ev	CMOVB, CMOVC, CMOVNAE Gv, Ev	CMOVAE, CM OVNB, CMOV NC Gv, Ev	CMOVE, CMOVZ Gv, Ev	CMOVNE, CMOVNZ Gv, Ev	CMOVBE, CMOVNA Gv, Ev	CMOVA, CMOVNBE Gv, Ev
5								
6	PUNPCKLBW Pq, Qd	PUNPCKLWD Pq, Qd	PUNOCKLDQ Pq, Qd	PACKUSDW Pq, Qd	PCM PGTB Pq, Qd	PCM PGTW Pq, Qd	PCM PGTD Pq, Qd	PACKSSW B Pq, Qd
			Group A ²			PCM PEQB Pq, Qd	PCM PEQW Pq, Qd	PCM PEQD Pq, Qd
7		PSHIMW ³	PSHIMD ³	PSHIMQ ³				EMMS
8	Long-Displacement Jump on Condition (Jv)							
	JO	JNO	JB	JNB	JZ	JNZ	JBE	JNBE
9	Byte Set on condition (Eb)							
	SETO	SETNO	SETB	SETNB	SETZ	SETNZ	SETBE	SETNBE
A	PUSH FS	POP FS	CPUID	BT Ev,Gv	SHLD Ev,Gv,Ib	SHLD Ev,Gv,CL		
B	CMPXCHG	CMPXCHG	LSS	BTR	LFS	LGS	MOVZX	
	Eb,Gb	Ev,Gv	Mp	Ev,Gv	Mp	Mp	Gv,Eb	Gv,Ew
C	XADD Eb,Gb	XADD Ev,Gv					Group 9 ²	
D		PSRLW Pq, Qd	PSRLD Pq, Qd	PSRLQ Pq, Qd		PMULLW Pq, Qd		
E		PSRAW Pq, Qd	PSRAD Pq, Qd			PMULHW Pq, Qd		
F		PSLLW Pq, Qd	PSLLD Pq, Qd	PSLLQ Pq, Qd		PMADDWD Pq, Qd		

Table 37-4. Two-Byte Opcode Map (8–F) (First byte is 0FH)

	8	9	A	B	C	D	E	F
0	INVD	WBINVD		UD2 ⁴				
1								
2								
3								
4	CMOVS Gv, Ev	CMOVNS Gv, Ev	CMOVP, CMOVPE Gv, Ev	CMOVNP, CMOVPO Gv, Ev	CMOVL, CMOVNGE Gv, Ev	CMOVGE, CMOVNL Gv, Ev	CMOVLE, CMOVNG Gv, Ev	CMOVG, CMOVNLE Gv, Ev
5								
6	PUNPCKHBW Pq, Qd	PUNPCKHWD Pq, Qd	PUNPCKHDQ Pq, Qd	PACKSSDW Pq, Qd			MOVD Pd, Ed	MOVQ Pq, Qq
7							MOVD Ed, Pd	MOVQ Qq, Pq
8	Long-Displacement Jump on Condition (Jv)							
	JS	JNS	JP	JNP	JL	JNL	JLE	JNLE
	Byte set on condition (Eb)							
9	SETS	SETNS	SETP	SETNP	SETL	SETNL	SETLE	SETNLE
	Eb	Eb	Eb	Eb	Eb	Eb	Eb	Eb
A	PUSH GS	POP GS	RSM	BTS Ev,Gv	SHRD Ev,Gv,Ib	SHRD Ev,Gv,CL		IMUL Gv,Ev
B		Invalid Opcode ⁴	Group 8 ²	BTC	BSF	BSR	MOVSX	
			Ev,Ib	Ev,Gv	Gv,Ev	Gv,Ev	Gv,Eb	Gv,Ew
C	BSWAP EAX	BSWAP ECX	BSWAP EDX	BSWAP EBX	BSWAP ESP	BSWAP EBP	BSWAP ESI	BSWAP EDI
D	PSUBUSB Pq, Qq	PSUBUSW Pq, Qq		PAND Pq, Qq	PADDUSB Pq, Qq	PADDUSW Pq, Qq		PANDN Pq, Qq
E	PSUBSB Pq, Qq	PSUBSW Pq, Qq		POR Pq, Qq	PADDSB Pq, Qq	PADDSW Pq, Qq		PXOR Pq, Qq
F	PSUBB Pq, Qq	PSUBW Pq, Qq	PSUBD Pq, Qq		PADDB Pq, Qq	PADDW Pq, Qq	PADDD Pq, Qq	

NOTES:

1. All blanks in the opcode map are reserved and should not be used. Do not depend on the operation of these undefined opcodes.
2. Bits 5, 4, and 3 of ModR/M byte used as an opcode extension (see “Opcode Extensions for One- and Two-byte Opcodes”).
3. These abbreviations are not actual mnemonics. When shifting by immediate shift counts, the PSHIMD mnemonic represents the PSLLD, PSRAD, and PSRLD instructions, PSHIMW represents the PSLLW, PSRAW, and PSRLW instructions, and PSHIMQ represents the PSLLQ and PSRLQ instructions. The instructions that shift by immediate counts are differentiated by the ModR/M bytes (see “Opcode Extensions for One- and Two-byte Opcodes”).
4. Use the 0F0B opcode (UD2 instruction) or the 0FB9H opcode when deliberately trying to generate an invalid opcode exception (#UD).

37.4 Opcode Extensions for One- and Two-byte Opcodes

Some of the 1-byte and 2-byte opcodes use bits 5, 4, and 3 of the ModR/M byte (the nnn field in Figure 37-5) as an extension of the opcode. Those opcodes that have opcode extensions are indicated in Tables 37-1 and 37-3 with group numbers (Group 1, Group 2, etc.). The group numbers (which range from 1 to A) provide an entry point into Table 37-6 where the encoding of the opcode extension field can be found. For example, the ADD instruction with a 1-byte opcode of 80H is a Group 1 instruction. Table 37-6 then indicates that the opcode extension that must be encoded in the ModR/M byte for this instruction is 000B.

Table 37-5. ModR/M Byte nnn Field (Bits 5, 4, and 3)

mod	nnn	R/M
-----	-----	-----

Table 37-6. Opcode Extensions for One- and Two-Byte Opcodes by Group Number¹

Group	Encoding of Bits 5,4,3 of the ModR/M Byte							
	000	001	010	011	100	101	110	111
1	ADD	OR	ADC	SBB	AND	SUB	XOR	CMP
2	ROL	ROR	RCL	RCR	SHL, SAL	SHR		SAR
3	TEST lb/lv		NOT	NEG	MUL AL/eAX	IMUL AL/eAX	DIV AL/eAX	IDIV AL/eAX
4	INC Eb	DEC Eb						
5	INC Ev	DEC Ev	CALL Ev	CALL Ep	JMP Ev	JMP Ep	PUSH Ev	
6	SLDT Ew	STR Ew	LLDT Ew	LTR Ew	VERR Ew	VERW Ew		
7	SGDT Ms	SIDT Ms	LGDT Ms	LIDT Ms	SMSW Ew		LMSW Ew	INVLPG
8					BT	BTS	BTR	BTC
9		CMPXCH8 BMq						
A			PSRLD, PSRLW, PSRLQ Pq, Ib		PSRAD, PSRAW Pq, Ib		PSLLD, PSLLW, PSLLQ Pq, Ib	

NOTE:

1. All blanks in the opcode map are reserved and should not be used. Do not depend on the operation of these undefined opcodes.

37.5 Escape Opcode Instructions

The opcode maps for the escape instruction opcodes (floating-point instruction opcodes) are given in Tables 37-7 through 37-16. These opcode maps are grouped by the first byte of the opcode from D8 through DF. Each of these opcodes has a ModR/M byte. If the ModR/M byte is within the range of 00H through BFH, bits 5, 4, and 3 of the ModR/M byte are used as an opcode extension, similar

to the technique used for 1-and 2-byte opcodes (see “Opcode Extensions for One- and Two-byte Opcodes”). If the ModR/M byte is outside the range of 00H through BFH, the entire ModR/M byte is used as an opcode extension.

For example, the opcode DD0504000000H can be interpreted as follows. The instruction encoded with this opcode can be located in “Escape Opcodes with DD as First Byte”. Since the ModR/M byte (05H) is within the 00H through BFH range, bits 3 through 5 (000) of this byte indicate the opcode to be for an FLD double-real instruction (see Table 37-9). The double-real value to be loaded is at 00000004H, which is the 32-bit displacement that follows and belongs to this opcode.

The opcode D8C1H illustrates an opcode with a ModR/M byte outside the range of 00H through BFH. The instruction encoded here, can be located in “Escape Opcodes with D8 as First Byte”. In Table 37-8, the ModR/M byte C1H indicates row C, column 1, which is an FADD instruction using ST(0), ST(1) as the operands.

37.5.1 Escape Opcodes with D8 as First Byte

Tables A-4 and A-5 contain the opcodes maps for the escape instruction opcodes that begin with D8H. Table A-4 shows the opcode map if the accompanying ModR/M byte within the range of 00H through BFH. Here, the value of bits 5, 4, and 3 (the nnn field in Figure 37-5) selects the instruction.

Table 37-7. D8 Opcode Map When ModR/M Byte is Within 00H to BFH¹

nnn Field of ModR/M Byte (see Figure 37-5)							
000	001	010	011	100	101	110	111
FADD single-real	FMUL single-real	FCOM single-real	FCOMP single-real	FSUB single-real	FSUBR single-real	FDIV single-real	FDIVR single-real

NOTE:

1. All blanks in the opcode map are reserved and should not be used. Do not depend on the operation of these undefined opcodes.

Table A-5 shows the opcode map if the accompanying ModR/M byte is outside the range of 00H to BFH. In this case the first digit of the ModR/M byte selects the row in the table and the second digit selects the column.

Table 37-8. D8 Opcode Map When ModR/M Byte is Outside 00H to BFH¹

	0	1	2	3	4	5	6	7
C	FADD							
	ST(0),ST(0)	ST(0),ST(1)	ST(0),ST(2)	ST(0),ST(3)	ST(0),ST(4)	ST(0),ST(5)	ST(0),ST(6)	ST(0),ST(7)
D	FCOM							
	ST(0),ST(0)	ST(0),ST(1)	ST(0),T(2)	ST(0),ST(3)	ST(0),ST(4)	ST(0),ST(5)	ST(0),ST(6)	ST(0),ST(7)
E	FSUB							
	ST(0),ST(0)	ST(0),ST(1)	ST(0),ST(2)	ST(0),ST(3)	ST(0),ST(4)	ST(0),ST(5)	ST(0),ST(6)	ST(0),ST(7)
F	FDIV							
	ST(0),ST(0)	ST(0),ST(1)	ST(0),ST(2)	ST(0),ST(3)	ST(0),ST(4)	ST(0),ST(5)	ST(0),ST(6)	ST(0),ST(7)
8	9	A	B	C	D	E	F	
C	FMUL							
	ST(0),ST(0)	ST(0),ST(1)	ST(0),ST(2)	ST(0),ST(3)	ST(0),ST(4)	ST(0),ST(5)	ST(0),ST(6)	ST(0),ST(7)
D	FCOMP							
	ST(0),ST(0)	ST(0),ST(1)	ST(0),T(2)	ST(0),ST(3)	ST(0),ST(4)	ST(0),ST(5)	ST(0),ST(6)	ST(0),ST(7)
E	FSUBR							
	ST(0),ST(0)	ST(0),ST(1)	ST(0),ST(2)	ST(0),ST(3)	ST(0),ST(4)	ST(0),ST(5)	ST(0),ST(6)	ST(0),ST(7)
F	FDIVR							
	ST(0),ST(0)	ST(0),ST(1)	ST(0),ST(2)	ST(0),ST(3)	ST(0),ST(4)	ST(0),ST(5)	ST(0),ST(6)	ST(0),ST(7)

NOTE:

1. All blanks in the opcode map are reserved and should not be used. Do not depend on the operation of these undefined opcodes.

37.5.2 Escape Opcodes with D9 as First Byte

Tables 37-9 and 37-16 contain the opcodes maps for the escape instruction opcodes that begin with D9H. Table 37-9 shows the opcode map if the accompanying ModR/M byte within the range of 00H through BFH. Here, the value of bits 5, 4, and 3 (the nnn field in Figure 37-5) selects the instruction.

Table 37-9. D9 Opcode Map When ModR/M Byte is Within 00H to BFH¹

nnn Field of ModR/M Byte (see Figure 37-5)							
000	001	010	011	100	101	110	111
FLD single-real		FST single-real	FSTP single-real	FLDENV 14/28 bytes	FLDCW 2 bytes	FSTENV 14/28 bytes	FSTCW 2 bytes

NOTE:

1. All blanks in the opcode map are reserved and should not be used. Do not depend on the operation of these undefined opcodes.

Table 37-16 shows the opcode map if the accompanying ModR/M byte is outside the range of 00H to BFH. In this case the first digit of the ModR/M byte selects the row in the table and the second digit selects the column.

Table 37-10. D9 Opcode Map When ModR/M Byte is Outside 00H to BFH¹

	0	1	2	3	4	5		7
C	FLD							
	ST(0),ST(0)	ST(0),ST(1)	ST(0),ST(2)	ST(0),ST(3)	ST(0),ST(4)	ST(0),ST(5)	ST(0),ST(6)	ST(0),ST(7)
D	FNOP							
E	FCHS	FABS			FTST	FXAM		
F	F2XM1	FYL2X	FPTAN	FPATAN	FXTRACT	FPREM1	FDECSTP	FINCSTP
	8	9	A	B	C	D	E	F
C	FXCH							
	ST(0),ST(0)	ST(0),ST(1)	ST(0),ST(2)	ST(0),ST(3)	ST(0),ST(4)	ST(0),ST(5)	ST(0),ST(6)	ST(0),ST(7)
D								
E	FLD1	FLDL2T	FLDL2E	FLDP1	FLDLG2	FLDLN2	FLDZ	

Table 37-10. D9 Opcode Map When ModR/M Byte is Outside 00H to BFH¹

F	FPREM	FYL2XP1	FSQRT	FSINCOS	FRNDINT	FSCALE	FSIN	FCOS

NOTE:

1. All blanks in the opcode map are reserved and should not be used. Do not depend on the operation of these undefined opcodes.

37.5.3 Escape Opcodes with DA as First Byte

Tables 37-9 and 37-16 contain the opcodes maps for the escape instruction opcodes that begin with DAH. Table 37-9 shows the opcode map if the accompanying ModR/M byte within the range of 00H through BFH. Here, the value of bits 5, 4, and 3 (the nnn field in Figure 37-5) selects the instruction.

Table 37-11. DA Opcode Map When ModR/M Byte is Within 00H to BFH¹

nnn Field of ModR/M Byte (see Figure 37-5)							
000	001	010	011	100	101	110	111
FIADD short-integer	FIMUL short-integer	FICOM short-integer	FICOMP short-integer	FISUB short-integer	FISUBR short-integer	FIDIV short-integer	FIDIVR short-integer

NOTE:

1. All blanks in the opcode map are reserved and should not be used. Do not depend on the operation of these undefined opcodes.

Table 37-16 shows the opcode map if the accompanying ModR/M byte is outside the range of 00H to BFH. In this case the first digit of the ModR/M byte selects the row in the table and the second digit selects the column.

Table 37-12. DA Opcode Map When ModR/M Byte is Outside 00H to BFH¹ (Sheet 1 of 2) (Sheet 1 of 2)

	0	1	2	3	4	5		7
C	FCMOV _B							
	ST(0),ST(0)	ST(0),ST(1)	ST(0),ST(2)	ST(0),ST(3)	ST(0),ST(4)	ST(0),ST(5)	ST(0),ST(6)	ST(0),ST(7)
D	FCMOV _{BE}							
	ST(0),ST(0)	ST(0),ST(1)	ST(0),ST(2)	ST(0),ST(3)	ST(0),ST(4)	ST(0),ST(5)	ST(0),ST(6)	ST(0),ST(7)
E								
F								
	8	9	A	B	C	D	E	F

Table 37-12. DA Opcode Map When ModR/M Byte is Outside 00H to BFH¹ (Sheet 2 of 2) (Sheet 2 of 2)

C	FCMOVE							
	ST(0),ST(0)	ST(0),ST(1)	ST(0),ST(2)	ST(0),ST(3)	ST(0),ST(4)	ST(0),ST(5)	ST(0),ST(6)	ST(0),ST(7)
D	FCMOVU							
	ST(0),ST(0)	ST(0),ST(1)	ST(0),ST(2)	ST(0),ST(3)	ST(0),ST(4)	ST(0),ST(5)	ST(0),ST(6)	ST(0),ST(7)
E	FUCOMPP							
F								

NOTE:

1. All blanks in the opcode map are reserved and should not be used. Do not depend on the operation of these undefined opcodes.

37.5.4 Escape Opcodes with DB as First Byte

Tables 37-9 and 37-16 contain the opcodes maps for the escape instruction opcodes that begin with DBH. Table 37-9 shows the opcode map if the accompanying ModR/M byte within the range of 00H through BFH. Here, the value of bits 5, 4, and 3 (the nnn field in Figure 37-5) selects the instruction.

Table 37-13. DB Opcode Map When ModR/M Byte is Within 00H to BFH¹

nnn Field of ModR/M Byte (see Figure 37-5)							
000	001	010	011	100	101	110	111
FILD short-integer		FIST short-integer	FISTP short-integer		FLD extended-real		FSTP extended-real

NOTE:

1. All blanks in the opcode map are reserved and should not be used. Do not depend on the operation of these undefined opcodes.

Table 37-16 shows the opcode map if the accompanying ModR/M byte is outside the range of 00H to BFH. In this case the first digit of the ModR/M byte selects the row in the table and the second digit selects the column.

Table 37-14. DB Opcode Map When ModR/M Byte is Outside 00H to BFH¹

	0	1	2	3	4	5		7
C	FCMOVNB							
	ST(0),ST(0)	ST(0),ST(1)	ST(0),ST(2)	ST(0),ST(3)	ST(0),ST(4)	ST(0),ST(5)	ST(0),ST(6)	ST(0),ST(7)
D	FCMOVNBE							
	ST(0),ST(0)	ST(0),ST(1)	ST(0),ST(2)	ST(0),ST(3)	ST(0),ST(4)	ST(0),ST(5)	ST(0),ST(6)	ST(0),ST(7)
E			FCLEX	FINIT				
F	FCOMI							
	ST(0),ST(0)	ST(0),ST(1)	ST(0),ST(2)	ST(0),ST(3)	ST(0),ST(4)	ST(0),ST(5)	ST(0),ST(6)	ST(0),ST(7)
8	9	A	B	C	D	E	F	
C	FCMOVNE							
	ST(0),ST(0)	ST(0),ST(1)	ST(0),ST(2)	ST(0),ST(3)	ST(0),ST(4)	ST(0),ST(5)	ST(0),ST(6)	ST(0),ST(7)
D	FCMOVNU							
	ST(0),ST(0)	ST(0),ST(1)	ST(0),ST(2)	ST(0),ST(3)	ST(0),ST(4)	ST(0),ST(5)	ST(0),ST(6)	ST(0),ST(7)
E	FUCOMI							
	ST(0),ST(0)	ST(0),ST(1)	ST(0),ST(2)	ST(0),ST(3)	ST(0),ST(4)	ST(0),ST(5)	ST(0),ST(6)	ST(0),ST(7)
F								

NOTE:

1. All blanks in the opcode map are reserved and should not be used. Do not depend on the operation of these undefined opcodes.

37.5.5 Escape Opcodes with DC as First Byte

Tables 37-9 and 37-16 contain the opcodes maps for the escape instruction opcodes that begin with DCH. Table 37-9 shows the opcode map if the accompanying ModR/M byte within the range of 00H through BFH. Here, the value of bits 5, 4, and 3 (the nnn field in Figure 37-5) selects the instruction.

Table 37-15. DC Opcode Map When ModR/M Byte is Within 00H to BFH¹

nnn Field of ModR/M Byte (see Figure 37-5)							
000	001	010	011	100	101	110	111
FADD double-real	FMUL double-real	FCOM double-real	FCOMP double-real	FSUB double-real	FSUBR double-real	FDIV double-real	FDIVR double-real

NOTE:

1. All blanks in the opcode map are reserved and should not be used. Do not depend on the operation of these undefined opcodes.

Table 37-16 shows the opcode map if the accompanying ModR/M byte is outside the range of 00H to BFH. In this case the first digit of the ModR/M byte selects the row in the table and the second digit selects the column.

Table 37-16. DC Opcode Map When ModR/M Byte is Outside 00H to BFH¹

	0	1	2	3	4	5		7
C	FADD							
	ST(0),ST(0)	ST(1),ST(0)	ST(2),ST(0)	ST(3),ST(0)	ST(4),ST(0)	ST(5),ST(0)	ST(6),ST(0)	ST(7),ST(0)
D								
E	FSUBR							
	ST(0),ST(0)	ST(1),ST(0)	ST(2),ST(0)	ST(3),ST(0)	ST(4),ST(0)	ST(5),ST(0)	ST(6),ST(0)	ST(7),ST(0)
F	FDIVR							
	ST(0),ST(0)	ST(1),ST(0)	ST(2),ST(0)	ST(3),ST(0)	ST(4),ST(0)	ST(5),ST(0)	ST(6),ST(0)	ST(7),ST(0)
	8	9	A	B	C	D	E	F
C	FMUL							
	ST(0),ST(0)	ST(1),ST(0)	ST(2),ST(0)	ST(3),ST(0)	ST(4),ST(0)	ST(5),ST(0)	ST(6),ST(0)	ST(7),ST(0)
D								
E	FSUB							
	ST(0),ST(0)	ST(1),ST(0)	ST(2),ST(0)	ST(3),ST(0)	ST(4),ST(0)	ST(5),ST(0)	ST(6),ST(0)	ST(7),ST(0)
F	FDIV							
	ST(0),ST(0)	ST(1),ST(0)	ST(2),ST(0)	ST(3),ST(0)	ST(4),ST(0)	ST(5),ST(0)	ST(6),ST(0)	ST(7),ST(0)

NOTE:

1. All blanks in the opcode map are reserved and should not be used. Do not depend on the operation of these undefined opcodes.

37.5.6 Escape Opcodes with DD as First Byte

Tables 37-9 and 37-16 contain the opcodes maps for the escape instruction opcodes that begin with DDH. Table 37-9 shows the opcode map if the accompanying ModR/M byte within the range of 00H through BFH. Here, the value of bits 5, 4, and 3 (the nnn field in Figure 37-5) selects the instruction.

Table 37-17. DD Opcode Map When ModR/M Byte is Within 00H to BFH¹

nnn Field of ModR/M Byte (see Figure 37-5)							
000	001	010	011	100	101	110	111
FLD double-real		FST double-real	FSTP double-real	FRSTOR 98/ 108bytes		FSAVE 98/ 108bytes	FSTSW 2 bytes

NOTE:

1. All blanks in the opcode map are reserved and should not be used. Do not depend on the operation of these undefined opcodes.

Table 37-16 shows the opcode map if the accompanying ModR/M byte is outside the range of 00H to BFH. In this case the first digit of the ModR/M byte selects the row in the table and the second digit selects the column.

Table 37-18. DD Opcode Map When ModR/M Byte is Outside 00H to BFH¹ (Sheet 1 of 2)

	0	1	2	3	4	5		7
C	FFREE							
	ST(0)	ST(1)	ST(2)	ST(3)	ST(4)	ST(5)	ST(6)	ST(7)
D	FST							
	ST(0)	ST(1)	ST(2)	ST(3)	ST(4)	ST(5)	ST(6)	ST(7)
E	FUCOM							
	ST(0),ST(0)	ST(1),ST(0)	ST(2),ST(0)	ST(3),ST(0)	ST(4),ST(0)	ST(5),ST(0)	ST(6),ST(0)	ST(7),ST(0)
F								
	8	9	A	B	C	D	E	F
C								
D	FSTP							
	ST(0)	ST(1)	ST(2)	ST(3)	ST(4)	ST(5)	ST(6)	ST(7)
E	FUCOMP							

Table 37-18. DD Opcode Map When ModR/M Byte is Outside 00H to BFH¹ (Sheet 2 of 2)

	0	1	2	3	4	5		7
	ST(0)	ST(1)	ST(2)	ST(3)	ST(4)	ST(5)	ST(6)	ST(7)
F								

NOTE:

1. All blanks in the opcode map are reserved and should not be used. Do not depend on the operation of these undefined opcodes.

37.5.7 Escape Opcodes with DE as First Byte

Tables 37-9 and 37-16 contain the opcodes maps for the escape instruction opcodes that begin with DEH. Table 37-9 shows the opcode map if the accompanying ModR/M byte within the range of 00H through BFH. Here, the value of bits 5, 4, and 3 (the nnn field in Figure 37-5) selects the instruction.

Table 37-19. DE Opcode Map When ModR/M Byte is Within 00H to BFH¹

nnn Field of ModR/M Byte (see Figure 37-5)							
000	001	010	011	100	101	110	111
FIADD word-integer	FIMUL word-integer	FICOM word-integer	FICOMP word-integer	FISUB word-integer	FISUBR word-integer	FIDIV word-integer	FIDIVR word-integer

NOTE:

1. All blanks in the opcode map are reserved and should not be used. Do not depend on the operation of these undefined opcodes.

Table 37-16 shows the opcode map if the accompanying ModR/M byte is outside the range of 00H to BFH. In this case the first digit of the ModR/M byte selects the row in the table and the second digit selects the column.

Table 37-20. DE Opcode Map When ModR/M Byte is Outside 00H to BFH¹

	0	1	2	3	4	5		7
C	FADDP							
D	ST(0),ST(0)	ST(1),ST(0)	ST(2),ST(0)	ST(3),ST(0)	ST(4),ST(0)	ST(5),ST(0)	ST(6),ST(0)	ST(7),ST(0)
E								
F	FSUBRP							
	ST(0),ST(0)	ST(1),ST(0)	ST(2),ST(0)	ST(3),ST(0)	ST(4),ST(0)	ST(5),ST(0)	ST(6),ST(0)	ST(7),ST(0)
	FDIVRP							
	ST(0),ST(0)	ST(1),ST(0)	ST(2),ST(0)	ST(3),ST(0)	ST(4),ST(0)	ST(5),ST(0)	ST(6),ST(0)	ST(7),ST(0)

Table 37-20. DE Opcode Map When ModR/M Byte is Outside 00H to BFH¹

	8	9	A	B	C	D	E	F
C	FMULP							
ST(0),ST(0)	ST(1),ST(0)	ST(2),ST(0)	ST(3),ST(0)	ST(4),ST(0)	ST(5),ST(0)	ST(6),ST(0)	ST(7),ST(0)	
D	FCOMPP							
E	FSUBP							
ST(0),ST(0)	ST(1),ST(0)	ST(2),ST(0)	ST(3),ST(0)	ST(4),ST(0)	ST(5),ST(0)	ST(6),ST(0)	ST(7),ST(0)	
F	FDIVP							
ST(0),ST(0)	ST(1),ST(0)	ST(2),ST(0)	ST(3),ST(0)	ST(4),ST(0)	ST(5),ST(0)	ST(6),ST(0)	ST(7),ST(0)	

NOTE:

1. All blanks in the opcode map are reserved and should not be used. Do not depend on the operation of these undefined opcodes.

37.5.8 Escape Opcodes with DF As First Byte

Tables 37-9 and 37-16 contain the opcodes maps for the escape instruction opcodes that begin with DFH. Table 37-9 shows the opcode map if the accompanying ModR/M byte within the range of 00H through BFH. Here, the value of bits 5, 4, and 3 (the nnn field in Figure 37-5) selects the instruction.

Table 37-21. DF Opcode Map When ModR/M Byte is Within 00H to BFH¹

nnn Field of ModR/M Byte (see Figure 37-5)							
000	001	010	011	100	101	110	111
FILD word-integer		FIST word-integer	FISTP word-integer	FBLD packed-BCD	FILD long-integer	FBSTP packed-BCD	FISTP long-integer

NOTE:

1. All blanks in the opcode map are reserved and should not be used. Do not depend on the operation of these undefined opcodes.

Table 37-16 shows the opcode map if the accompanying ModR/M byte is outside the range of 00H to BFH. In this case the first digit of the ModR/M byte selects the row in the table and the second digit selects the column.

Table 37-22. DF Opcode Map When ModR/M Byte is Outside 00H to BFH¹

	0	1	2	3	4	5		7
C								
D								

Table 37-22. DF Opcode Map When ModR/M Byte is Outside 00H to BFH¹

E	FSTSW AX							
F	FCOMIP							
	ST(0),ST(0)	ST(0),ST(1)	ST(0),ST(2)	ST(0),ST(3)	ST(0),ST(4)	ST(0),ST(5)	ST(0),ST(6)	ST(0),ST(7)
	8	9	A	B	C	D	E	F
C								
D								
E	FUCOMIP							
	ST(0),ST(0)	ST(0),ST(1)	ST(0),ST(2)	ST(0),ST(3)	ST(0),ST(4)	ST(0),ST(5)	ST(0),ST(6)	ST(0),ST(7)
F								

NOTE:

1. All blanks in the opcode map are reserved and should not be used. Do not depend on the operation of these undefined opcodes.